Journal of Integrated Sciences
Special Issue, February 2026
ISSN: 2806-4801

EFFICIENT RANSOMWARE DETECTION USING ENSEMBLE CLASSIFIERS

Sharmin Akter

Department of Computer Science and Engineering
City University

ABSTRACT

Ransomware is one of the most widespread and damaging threats in cybersecurity. These malicious applications encrypt valuable user data using powerful cryptographic algorithms, rendering it unobtainable until payment of a ransom is made, typically by way of cryptocurrencies such as Bitcoin. Signature-based detection methods are slow in terms of reaction time, and they cannot accurately deal with the new or zero-day ransomware strains. To address these problems, this paper introduces and evaluates an ensemble machine learning classifier trained on static features collected from Windows Portable Executable (PE) files. The model was trained and tested on a dataset consisting of around 20,000 instances, each with 134 static attributes. The experimental results prove that the model proposed in this work achieves a high detection accuracy of 98.80%, and is characterized by an extremely low false positive rate (0.26%) together with a very high true positive rate (97.08%). These observations also demonstrate that static-based ensemble classifiers are an effective approach to establish a proactive, lightweight, and scalable defense against modern ransomware attacks.

Keywords: *Ransomware, machine learning, gradient boosting, classification, cybersecurity.*

Corresponding author: Sharmin Akter can be contacted at sharmin.cse051@gmail.com

1. INTRODUCTION

Ransomware has been one of the most serious cybersecurity threats to emerge in recent years, impacting large segments of the economy such as healthcare, financial services and education with widespread operational interruption. Contemporary ransomware variants encrypt user files automatically and demand payment for the decryption key; however, the pressure is also psychological as it frustrates victims to use alternative ways of obtaining this information instead. Currently, more advanced and multiple attacks are launched against them, which are beyond the capability of signature-based and rule-based-detection (Zapata Sandoval et al., 2025). The prevalence of obfuscation, polymorphism and fileless execution in ransomware has led to the development of adaptive and intelligent detection methods.

Ransomware has become one of the most widespread and destructive categories of malware, affecting critical infrastructure, businesses and personal computing environments tens of millions of times per month. Unlike regular malware, ransomware is used to encrypt user data and demand a payment in return for recovery of the information making it operational or financially impossible. More modern genetic ransomwares use creative techniques to avoid trace detection by traditional network-defense mechanisms, which are generally signature-based and unable to detect the latest polymorphic and fileless ransomware. As such, we have seen increased interest in the use of machine learning (ML) and deep

learning (DL) models for detecting ransomware due to their capacity to model complex behavioral aspects and generalize across previously unseen patterns of attack.

A number of recent approaches have been introduced ranging from static code analysis approaches to dynamic behavior monitoring-based techniques and combining/semi-supervised ones (Routray et al. 2023). For example, the high detection rate of dynamic analysis models like EldeRan is attributed to analyzing early execution behaviors such as API calls and registry accesses (Sgandurra et al. 2016). As a result, static feature-based methods such as using opcode density/entropy/metadata, have shown high performance under reduced computation complexity (Ashraf A et al. 2019). Deep learning approaches such as CNNs, LSTMs, or hybrid architectures have advanced detection further, especially considering difficult environments like IoT devices and industrial control systems. Moreover, new detection techniques relying on side-channel signals (e.g., CPU usage and energy consumption) have been proposed that provide stealthier, lightweight solutions for resource-constrained scenarios.

2. REVIEW OF LITERATURE

Reinforced by its capability to learn the behavior of ransomware, detect zero-day threats and generalize over non-observed samples, machine learning (ML) or deep learning (DL) are becoming promising techniques for detecting the presence of ransomware. Early dynamic analysis techniques, such as EldeRan also utilized runtime indicators early in their implementation: registry modifications or API calls, to achieve an AUC of 0.995 (Sgandurra et al., 2016). Static analysis

methods, such as opcode density, and entropy-based categorization, have similarly shown to deliver high accuracy at low resource cost (Baldwin & Dehghantanha, 2018; Chaudhary & Adhikari, 2024). Newer research has investigated hybrid models of both static and dynamic features, deep neural networks (Ashraf et al., 2019), and kernel-level telemetry using eBPF (Brodzik et al., 2024), improving detection accuracy.

In addition, researchers have started to address ransomware detection in dedicated environments such as IoT (Sharma et al., 2023; Duhayyim et al., 2023), SCADA systems (Basnet et al., 2021), and mobile world (Kritika, 2024). In these directions, lightweight models and side-channel analysis (using CPU or energy consumption) approaches (Larsen et al., 2021; Niyaz et al., 2018) have demonstrated positive outlooks. There are some limitations, though, such as real-time detection, adversarial robustness, explainable, and resource-efficient (Yang & Sahita, 2020; Zhu et al., 2021).

Ransomware detection has been greatly improved by machine learning (ML) methods and deep learning (DL) techniques. Early works, such as EldeRan from Sgandurra et al. (2016) used API calls, coupled with registry activity to create dynamic behavioral features, and achieved nearly perfect AUC scores (~0.995). Static analysis approaches also started to become popular, e.g., Baldwin and Dehghantanha (2018) used opcode density for binary detection and classification with 100% precision rate in ransomware and benign files. Hybrid architectures which are comprised of static and dynamic features as in Ashraf et al. (2019), better detection rates through CNNs combined with classic classifiers. Deep Learning approaches such as LSTM and

DNNs achieve a high detection rate for ransomware in SCADA system (Basnet et al., 2021) and Android ecosystem (Kritika, 2024). Innovative methods like exploitation of CPU sensor data (Larsen et al., 2021) and power consumption curves (Niyaz et al., 2018) brought side-channel analysis research for early detection. Recent efforts have centered on ransomware in IoT, as Sharma et al. (2023) and Duhayyim et al. (2023) used lightweight ML and deep belief networks which were enhanced by metaheuristics, with more than 95% accuracy. Researchers are increasingly aware of adversarial robustness and explainability, with Yang & Sahita (2020) treating classifier resiliency and Zhu et al. (2021) with Siamese networks for few-shot learning. In addition, kernel-level ML is available based on eBPF (Brodzik et al., 2024) for early and stealthy detection in OS itself. Comparative studies (Zapata Sandoval et al., 2025; Kritika, 2024) and benchmark evaluations show that models from the ML-DL family can be successful in practice, but their effectiveness is significantly affected by feature selection process, dataset quality and also dependent on the running environment. A majority of models across the board achieve high accuracy (>95%) but struggle with changes in ransomware tactics, generality, and real-time applicability in resource limited environments.

3. RESEARCH METHODOLOGY

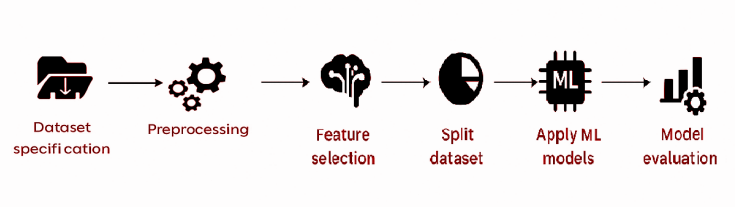

Figure 1. Methodology diagram

Figure 1 depicts stages in a machine learning process. It starts with dataset specification, where the data of interest is identified and collected. The data undergoes preprocessing to clean the data and convert it into a suitable format. Feature selection is performed to determine the most important variables for model training. The dataset is split into training and test sets to evaluate the performance of the model adequately. Different machine learning algorithms are trained on the training data, and finally, the models are tested with the right performance measures to see how well they perform.

3.1 Dataset specification

We retrieved this information from the virusshare.com website and benign PE files in the Windows OS. Then we will use advanced machine learning methods, such as AdaBoost, Gradient Boosting, Decision Tree, and Random Forest, in order to formulate classification models of the PE files - whether they are malware or not -, given only the majority of variables occurring in this database about their PE headers. The experiment is conducted to seek out a machine learning classifier that has a high accuracy and doesn't take long to be

trained. The dataset involves approximately 20,000 instances represented with 134 static attributes. The effectiveness of the detection model depends on the quality of the dataset. But high dimensions data that amplifies noise is one of the factors contributing to a challenging detection model.

3.2 Preprocessing

Data pre-processing is an important step in ML as it improves the quality of data and helps to gain some really useful insights from datasets. Accordingly, we performed the following pre-processing of data in our dataset. Data cleaning, whereby the duplicate samples are discarded after they have been detected through analysis.

3.3 Feature Selection

Correlational study is a research design that investigates the relationships between two or more sets of variables. It is one of the fundamental feature selections and easy to operate. When we have a pair of variables, in which the behavior of one variable has an effect on another variable is called correlation. Correlations could be both strong or weak, positive or negative. Not that there necessarily should be.

3.4 Split Dataset

The dataset is split into training and testing (and sometimes validation) sets. This allows the model to train on a portion of the data and be evaluated on data it hasn't seen before. Figure 2 shows the portion of testing and training split of the dataset.

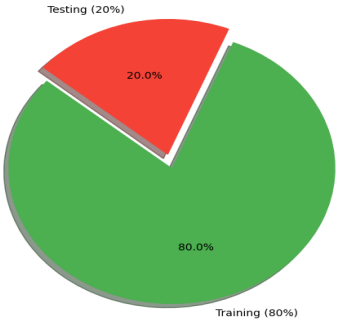

Figure 2. Dataset split

3.5 Applied ML Algorithms

The following ML techniques have been employed for detection.

XGboost: A fast and scalable tree boosting system outperforming all existing single-algorithm methods in almost all classification problems. It is designed to increase efficiency with minimal cost and computational time. It is designed primarily for improving the performance and computational speed of machine learning models. Trees are constructed in parallel instead of sequentially by XGBoost. Using partial sums, it applies a level-wise strategy to scan across gradient values and evaluate the goodness of split at each potential split on the partitioned dataset in training.

LightGBM: This is a discrete modulator on decision trees as a gradient boosting framework and increases the efficiency of the model with low memory requirements.

AdaBoost: AdaBoost is a Boosting algorithm used as an ensemble in machine learning. Weights are assigned to instances such that instances which were misclassified have higher weights (that is why it was called “adaptive boosting”). In supervised learning, boosting is intended to reduce the bias and the variance while preserving higher-order statistics. It is based on the assumption that students’ progress in a series of stages.

Random forest (RF) (Sruthi ER. 2023): Random Forest algorithm is an ensemble method known as Bootstrap Aggregation or bagging. This is where you randomly choose a subset of your data set to build these models over and over again. The two models are trained on unique subsets of the original data, referred to as Bootstrap Samples. The overall result is the combined output of all models, often computed as a majority vote or an average of results.

3.6 Model Evaluation

The comparison between the models based on the True Positive Rate (TPR) and False Positive Rate (FPR) properly describes each model's ability to label correctly positive cases and prevent false alarms. Higher TPR implies better sensitivity or recall, indicating that the model performs well in having the correct positive predictions. Lower FPR implies better specificity, that the model gives fewer incorrect positive predictions.

4. RESULTS AND DISCUSSION

4.1 Experiment Result

Model comparison the [table 1] reveals that XGBoost is the most appealing one among four methods in terms of overall (on and off-label) performance. It achieves a maximum accuracy of 98.80%, F1-score=98.78%, precision=98.77%, and recall = 98.77% with TPR = 97.08% and the minimum FPR is only equal to 0.26%. This demonstrates that XGBoost is not only able to detect positive cases with high accuracy, but it also can avoid giving many false alarms, which is the most trusted result model in our test. LightGBM and AdaBoost fare well, with the accuracy of 97.26% and 97.56%, respectively, but both have a slightly lower recall rate as well as FPR than XGBoost. Conversely, the performance of the Random Forest model is evidently inferior to XGBoost and Catboost by recording an accuracy of 91.28%, F1-score 89.11% with higher FPR of 0.74%), indicative of poor classification efficacy and potential underfitting/under-parameter tuning. In a word, the findings obviously demonstrate that XGBoost is the best and most balanced classifier since it's excellent generalizability as well as robustness in distinguishing true positive with low false positive.

Table 1. Performance summary of applied ML algorithms

Model	Train score	Accuracy	F1	Precision	Recall	TPR	FPR
XGBoost	100	98.80	98.78	98.77	98.77	97.08	0.26
LightGBM	100	97.26	97.20	97.16	97.18	95.22	0.31
AdaBoost	100	97.56	96.78	96.71	96.05	94.04	0.30
Random Forest (RF)	97.13	91.28	89.11	89.19	89.20	88.97	0.74

Source: The author's own work.

4.2 Discussion

Comparison to four ensemble learning methods (XGBoost, LightGBM AdaBoost and Random Forest) Figure 3 presents the respective classification performance differences of four ensemble learning models. Of these, XGBoost performs well in predictive power such that it obtains the best accuracy (98.80%), F1-score (98.78%), precision (98.77%) and recall (98.77%). The true positive rate (TPR) is 97.08%, and the false positive rate (FPR) is only 0.26% according to the model, which demonstrates superior sensitivity and robustness. This level of performance indicates that XGBoost optimally trades precision and recall, decreasing both false negatives and false positives; meanwhile this is important in sensitive classification studies that require a high detection rate.

Similarly, LightGBM and AdaBoost can achieve comparable performance (97.26% for LightGBM, 97.56% for AdaBoost). Nevertheless, the lower recall and higher false positive compared to XGBoost suggest a little loss in sensitivity.

Although both models achieve a perfect training accuracy, at the test level, their typically strikingly less better performance might be attributed to some degree of overfitting. However, the strong generalization ability on unseen data demonstrates their practicality for real-world applications with proper regularization.

On the other hand, the Random Forest model drops remarkably in performance with an accuracy of 91.28% and F1-score of 89.11%. Its FPR (0.74) is also larger than other algorithms, which indicates less discriminative power and generalization ability. It is clear that, for the Random Forest model, there might be some underfitting or hyperparameters, like the number of estimators or tree depth can still be improved.

In general (XGboost), it can be read out that XGboost works best for this dataset, and is able to find a good compromise between sensitivity and specificity. The good performance of its results on all evaluation metrics indicates the capability to capture nonlinear feature relationships and solve imbalance data problems effectively. These results are in accordance with the existing works, which demonstrate how XGBoost can lead to better accuracy and stability over different machine learning tasks given its gradient boosting framework and regularization techniques. In future work, the model interpretability can be improved using SHAP or feature importance analysis, along with a hybrid approach such as XGBoost and deep learning models for better predictive performance.

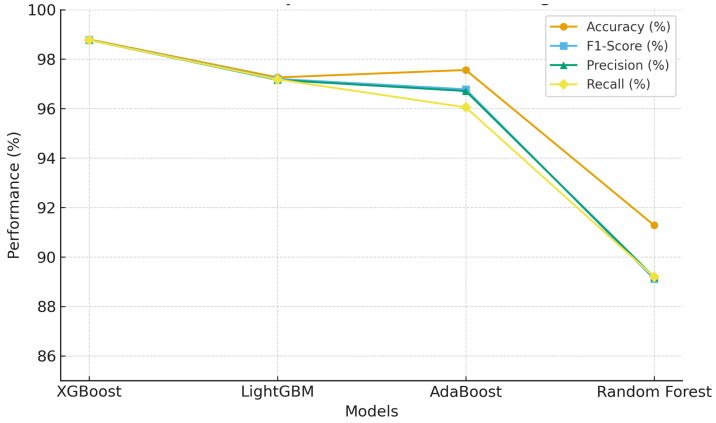

Figure 3. Performance comparison of applied ML models

5. CONCLUSION AND FUTURE DIRECTIONS

Four ensemble learning models (XGBoost, LightGBM, AdaBoost, and Random Forest) were considered in the present study, which were benchmarked on a performance analysis including accuracy, F1 score, precision, recall, true positive rate (TPR), and false positive rate(FPR). Here, XGBoost effectively balances sensitivity and specificity, providing reliable and robust classification outcomes. XGBoost is identified as the most effective model on this dataset, capable of both high precision and fewer false predictions. The strong generalization capability and the stability of the proposed system can make it practical for classifying real-world datasets.

Future work can follow several directions to further this research. First of all, hybrid architectures involving the fusion of ensemble learning and deep neural networks are introduced for

enhancing detection quality and robustness. Second, to evaluate the scalability and adaptability of models by experimenting on larger and heterogeneous datasets using cross-domain validation. Lastly, the investigation for adversarial robustness and real-time performance optimization can further improve the model's suitability in dynamic and security-critical scenarios.

REFERENCES

Sgandurra, D., Muñoz-González, L., Mohsen, R., & Lupu, E. C. (2016). *Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection.* arXiv.

Baldwin, J., & Dehghantanha, A. (2018). *Leveraging Support Vector Machine for Opcode Density Based Detection of Crypto-Ransomware.* arXiv.

Chen, L., Yang, C. Y., Paul, A., & Sahita, R. (2018). *Towards Resilient Machine Learning for Ransomware Detection.* arXiv.

Ashraf, A., Aziz, A., Zahoora, U., Rajarajan, M., & Khan, A. (2019). *Ransomware Analysis using Feature Engineering and Deep Neural Networks.* arXiv.

Larsen, E., Noever, D., & MacVittie, K. (2021). *A Survey of Machine Learning Algorithms for Detecting Ransomware Encryption Activity.* arXiv.

Basnet, M., Poudyal, S., Ali, M. H., & Dasgupta, D. (2021). *Ransomware Detection Using Deep Learning in the SCADA System of Electric Vehicle Charging Station.* IEEE / arXiv.

Duhayyim, M. A., Mohamed, H. G., Alrowais, F., Al-Wesabi, F. N., Hilal, A. M., & Motwakel, A. (2023). *Artificial Algae Optimization with Deep Belief Network Enabled Ransomware Detection in IoT Environment.* Computer Systems Science and Engineering, 46(2), 1293-1310.

Sharma, P., Kapoor, S., & Sharma, R. (2023). *Ransomware detection, prevention and protection in IoT devices using ML*

techniques based on dynamic analysis approach. International Journal of System Assurance Engineering and Management, 14(1), 287-296.

Ransomware Attack Detection on the Internet of Things Using Machine Learning Algorithm. (2022). HCI International Late Breaking Papers.

Niyaz, Q., Sun, W., & Alam, M. (2018). *Detecting crypto-ransomware in IoT networks based on energy consumption footprint*. Journal of Ambient Intelligence and Humanized Computing, 9, 1141-1152.

Chaudhary, I., & Adhikari, S. (2024). *Ransomware Detection Using Machine Learning Techniques*. Researcher CAB: A Journal for Research and Development, 3(1), 96-114.

Routray, S., Prusti, D., & Rath, S. K. (2023). *Ransomware Attack Detection by Applying Machine Learning Techniques*. In Machine Intelligence Techniques for Data Analysis and Signal Processing (Vol. 997, pp. 765-776). Springer.

Zhu, J., Jang-Jaccard, A., Singh, A., Welch, I., Al-Sahaf, H., & Camtepe, S. (2021). *A Few-Shot Meta-Learning based Siamese Neural Network using Entropy Features for Ransomware Classification*. arXiv:2112.00668.

Brodzik, A., Malec-Kruszyński, T., Niewolski, W., Tkaczyk, M., Bocianiak, K., & Loui, S-Y. (2024). *Ransomware Detection Using Machine Learning in the Linux Kernel*. (Preprint / PapersWithCode).

A Review on Android Ransomware Detection Using Deep Learning Techniques. (2019). In Proceedings of the 11th International Conference on Management of Digital EcoSystems (MEDES'19).

Yang, C. Y., Sahita, R., et al. (2020). *Towards a Resilient Machine Learning Classifier — a Case Study of Ransomware Detection*. arXiv.

Kritika (2024). *A comprehensive literature review on ransomware detection using deep learning*. Cyber Security and Applications, 3, 100078.

Zapata Sandoval, J. I., Garcés, E., & Fuertes, W. (2025). *Ransomware Detection with Machine Learning: Techniques, Challenges, and Future Directions – A Systematic Review*. Journal of Internet Services and Information Security, 15(1), 271-287.

Shaikh, M. R., Hassan, M. F., Akbar, R., Savita, K. S., Ullah, R., & Mandala, S. (2024). *Ransomware Classification with Deep Neural Network and Bi-LSTM*. Journal of Advanced Research in Applied Sciences and Engineering Technology, 47(2), 266-280.

Riaz, S., Latif, S., Usman, S. M., Ullah, S. S., Algarni, A. D., Yasin, A., Anwar, A., Elmannai, H., & Hussain, S. (2022). *Malware Detection in Internet of Things (IoT) Devices Using Deep Learning*. Sensors, 22(23), 9305.

Sruthi ER. Understand random forest algorithms with examples (updated 2023), 2023.